

Securing Microservices w/ Identity Server 4 and Ocelot



Mehmet Ozkaya

Software Architect, .NET

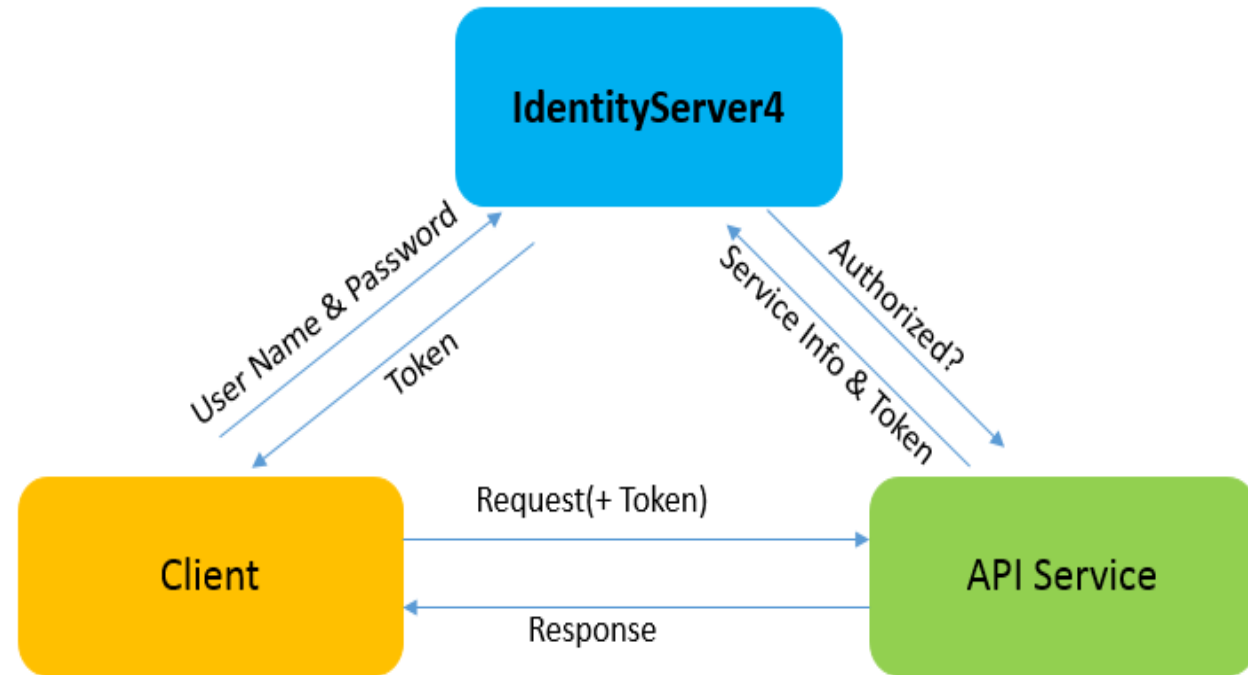
@ezozkme



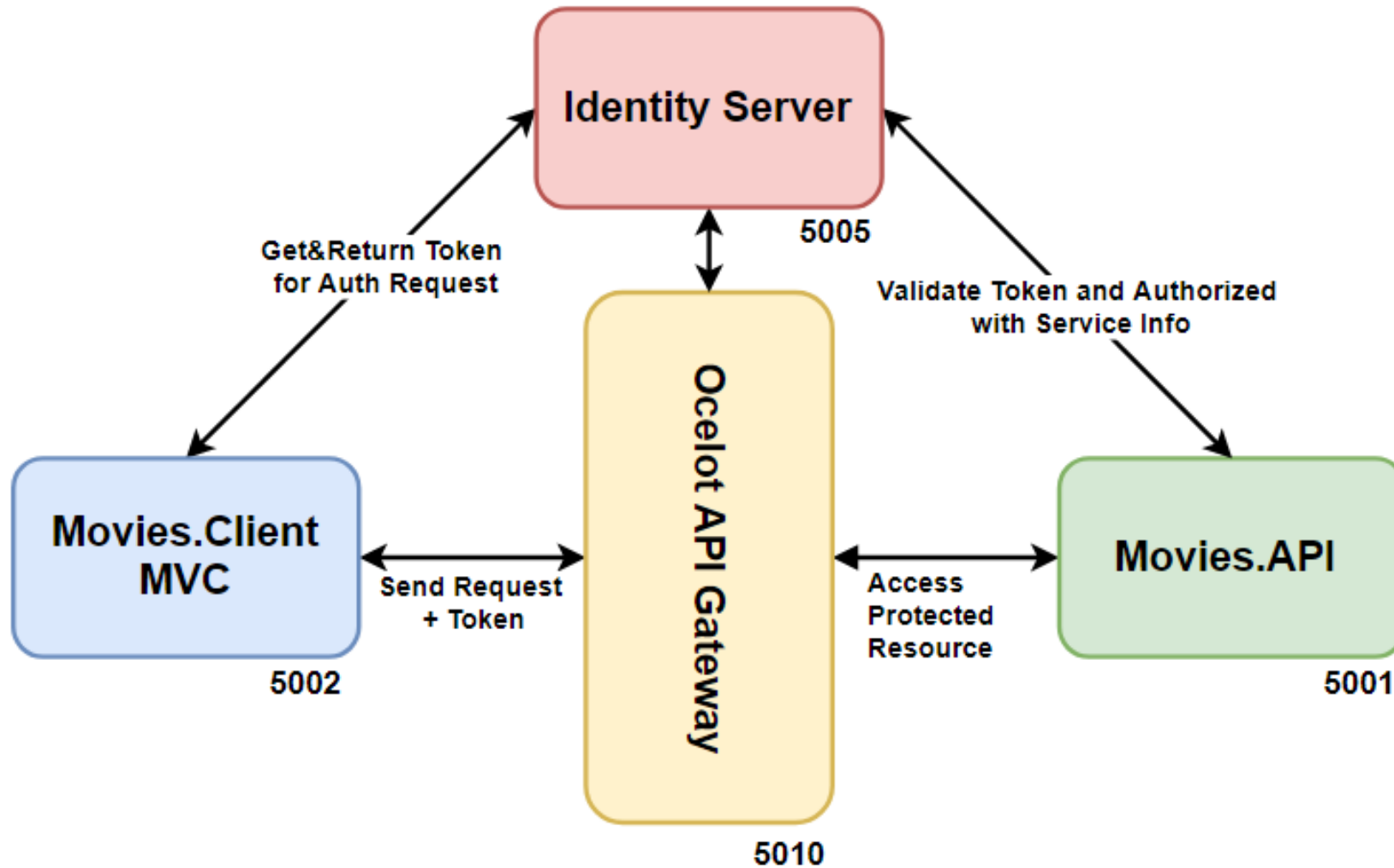
identity
SERVER

Microservices Security with IdentityServer4

- IdentityServer4 - Authentication as a Service
- Protected API Resources with OAuth 2.0
- OpenID Connect for MVC Interactive Client Apps
- Backing with Ocelot API Gateway



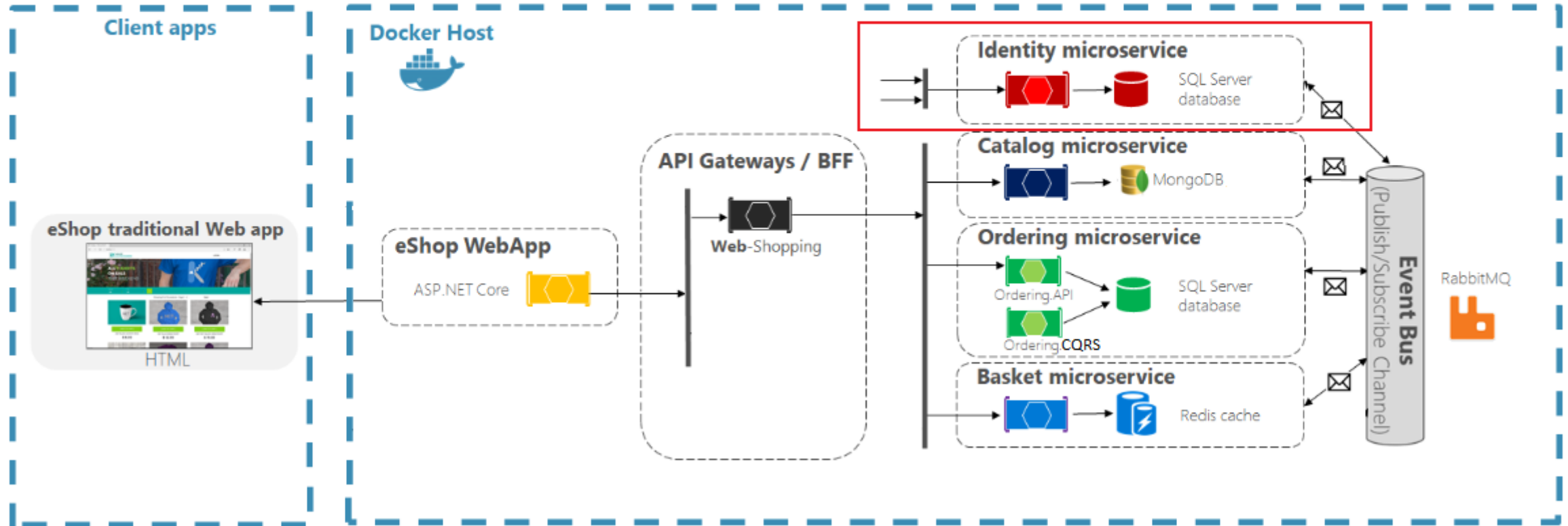
Big Picture



Identity Server in Microservices Architecture



aspnetrun-microservices Environment Architecture



Prerequisites



- Good knowledge of C#
- Have knowledge of Asp.Net MVC and REST API's
- No need to know about Oauth, OpenID Connect nor Identity Server 4
- Basics of Microservices Architecture

Existing Microservices Project

Microservices Architecture and Implementation on .NET

Building Microservices on .Net which used Asp.Net Web API, Docker, RabbitMQ, Ocelot API Gateway, MongoDB, Redis, SqlServer

★★★★★ 0.0 (0 ratings) 0 students enrolled

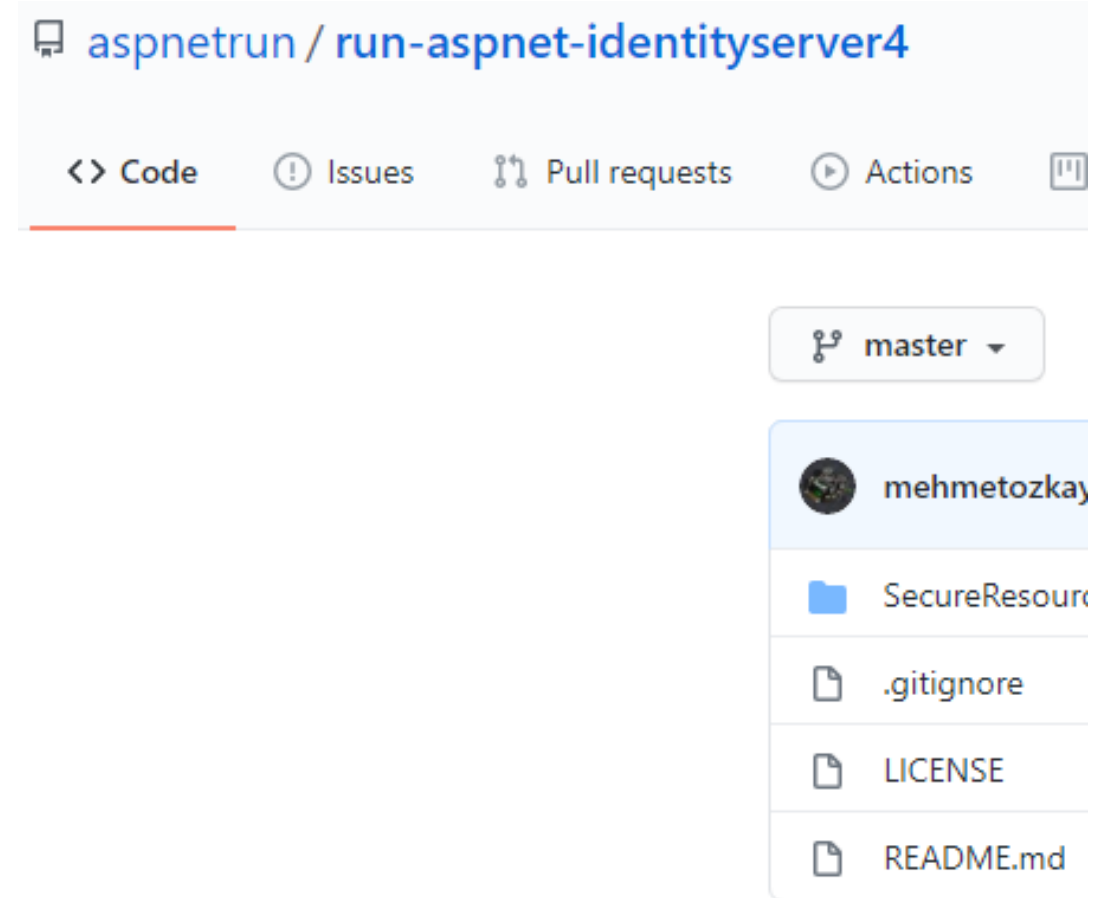
Created by Mehmet Özkaya Published 6/2020  English  English [Auto]

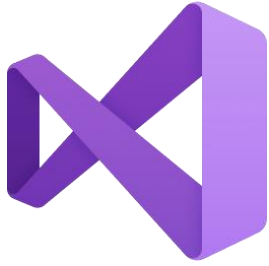
- IdentityServer4 implementation
- Github Repository -> <https://github.com/aspnetrun/run-aspnetcore-microservices>

Source Code on Github

- Source code on Github
- Fork the repository
- Open issues
- Send Pull Requests

<https://github.com/aspnetrun/run-aspnet-identityserver4>





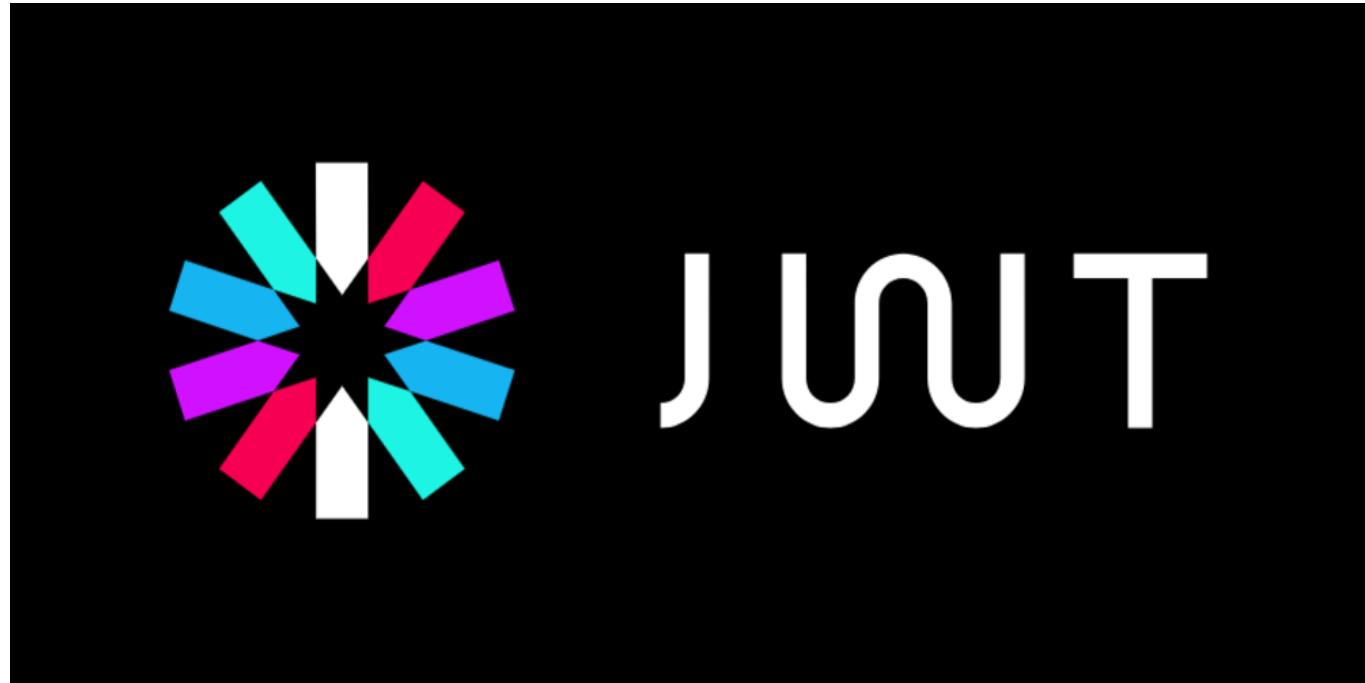
POSTMAN

Tooling

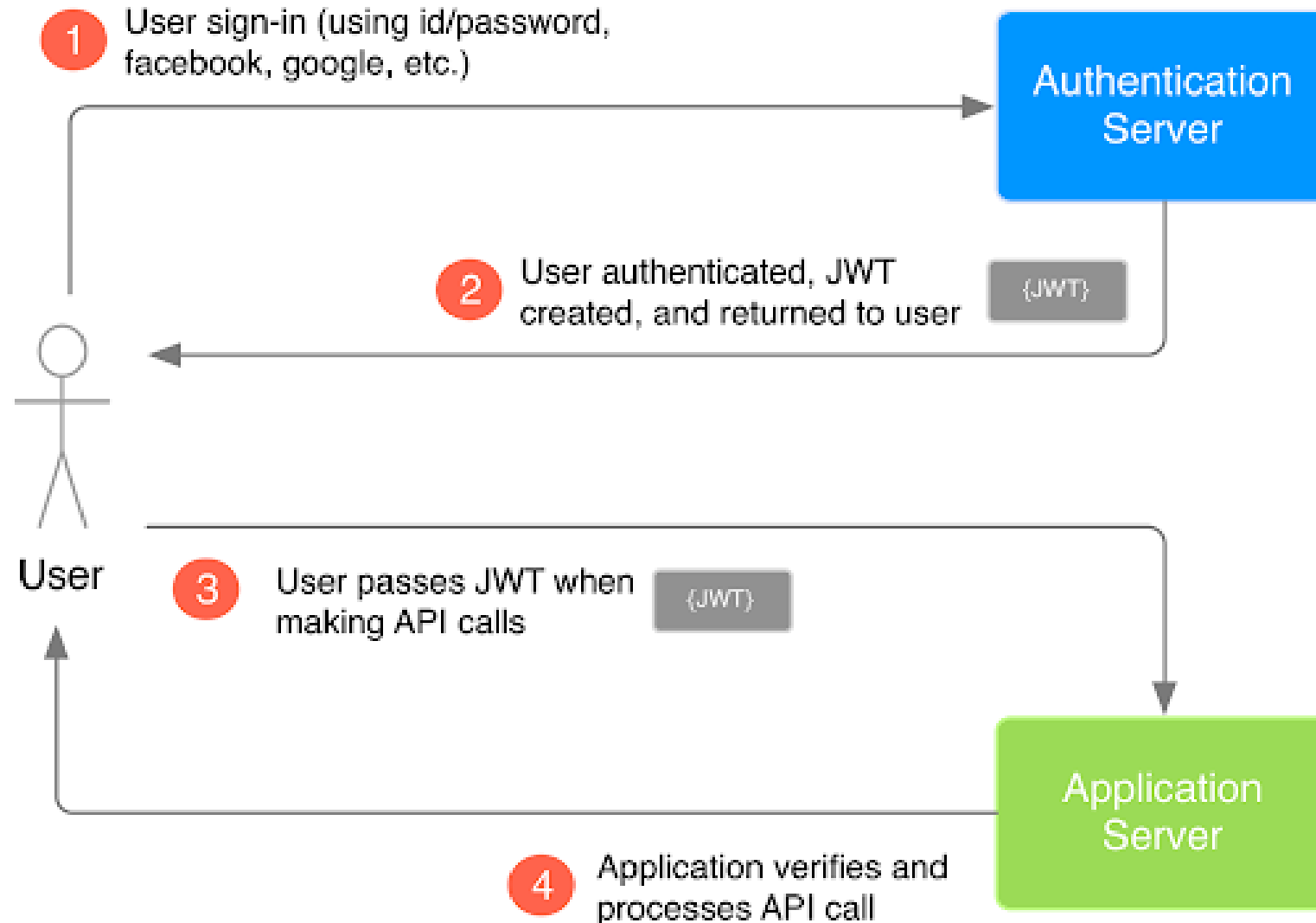
- Visual Studio 2019
- Postman
- Docker

What is JWT (Token) ?

- For Authentication and Authorization
- RFC7519 industry standard
- Web service security
- Retrieve a token
- <https://tools.ietf.org/html/rfc7519>



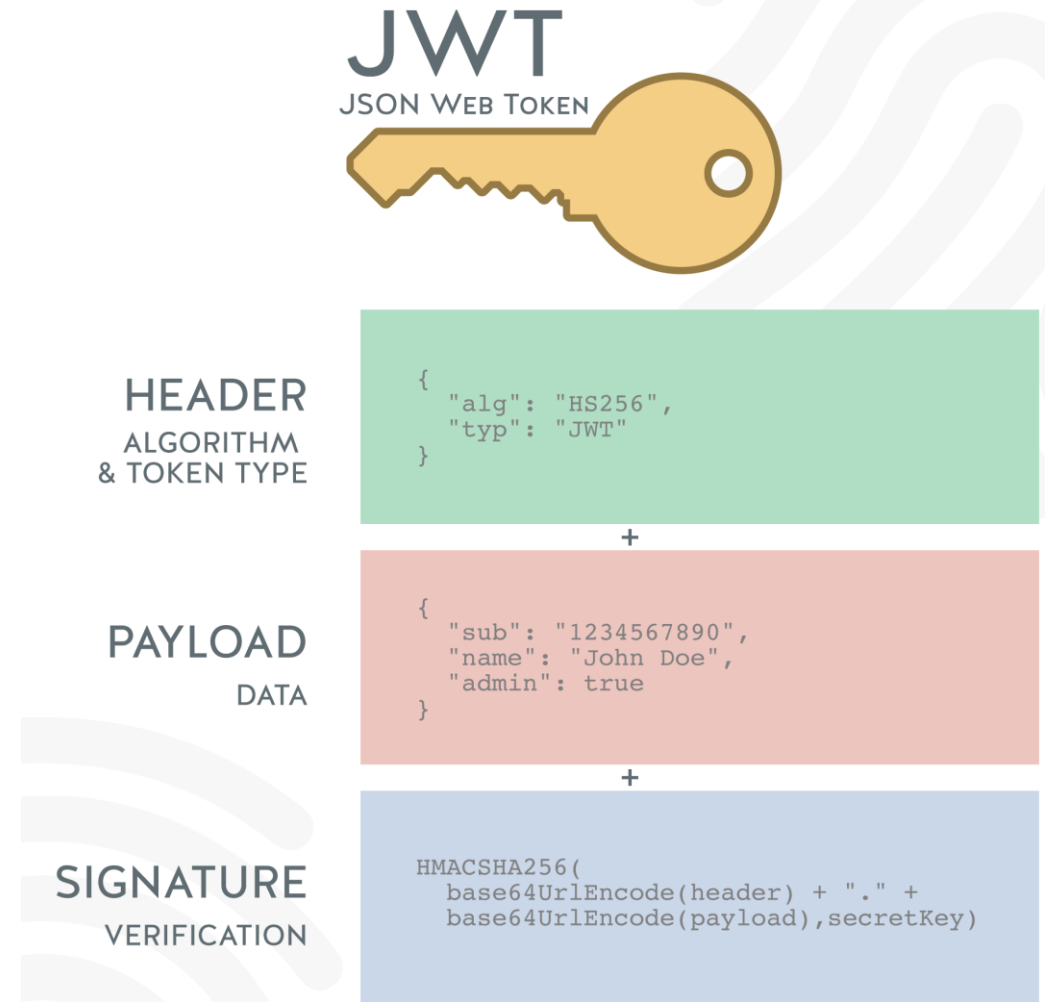
JWT Example Scenario



JWT(JSON Web Tokens) Structure

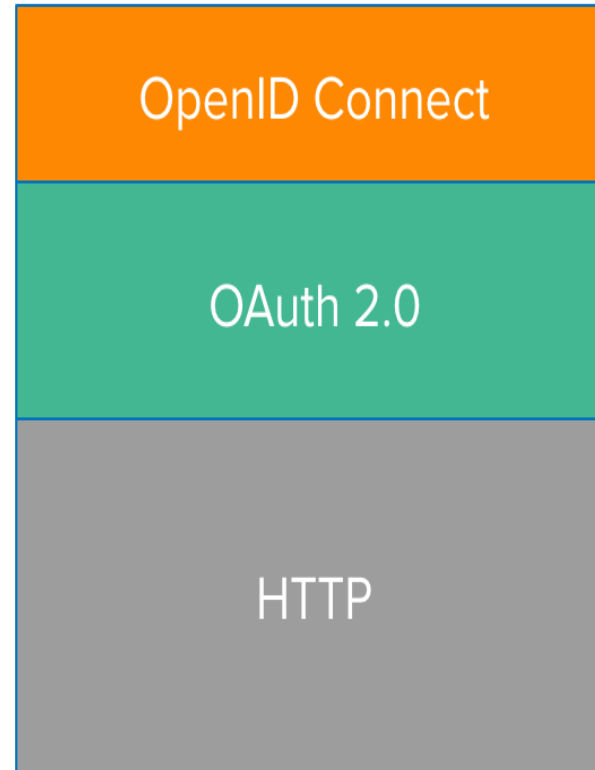
- Encoded with Base64
- Header
- Payload
- Signature

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.XbPfbIHM I6arZ3Y922BhjWgQzWXcXNrZ0ogtVhfEd2o



What is OAuth2 ?

- Authorization protocol
- [RFC 6749](https://tools.ietf.org/html/rfc6749) industry standard
- Web service security
- Third-party application access
- <https://tools.ietf.org/html/rfc6749>



OpenID Connect is for **authentication**

OAuth 2.0 is for **authorization**

OAuth2 Grant Types and Flows

- [RFC 6749](#) document
- Authorization Code Grant
- Implicit Grant
- Resource Owner Password Credential Grant
- Client Credential Grant

1.2. Protocol Flow

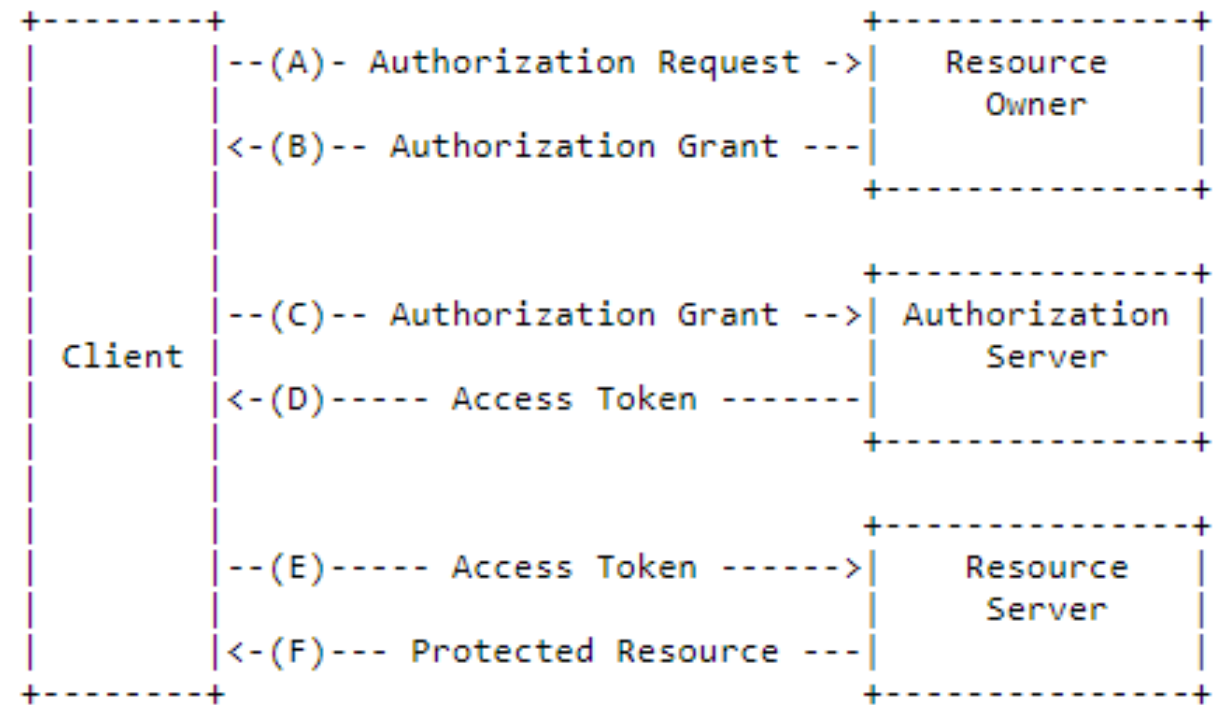
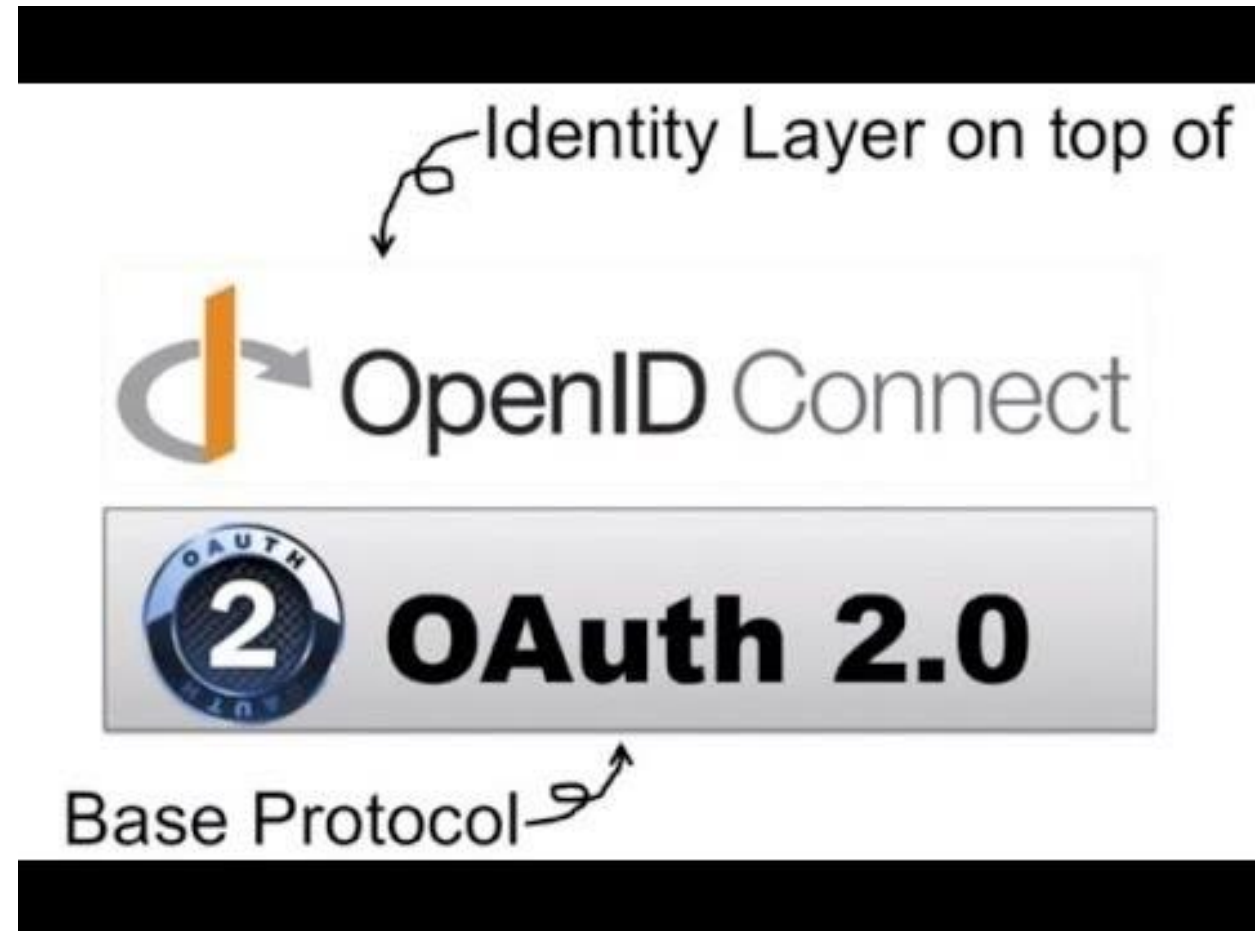


Figure 1: Abstract Protocol Flow

What is OpenID Connect ?

- Simple Authentication Layer
- Built on the OAuth2
- API friendly structure
- OAuth2 Authorization Code Grant
- https://openid.net/specs/openid-connect-core-1_0.html



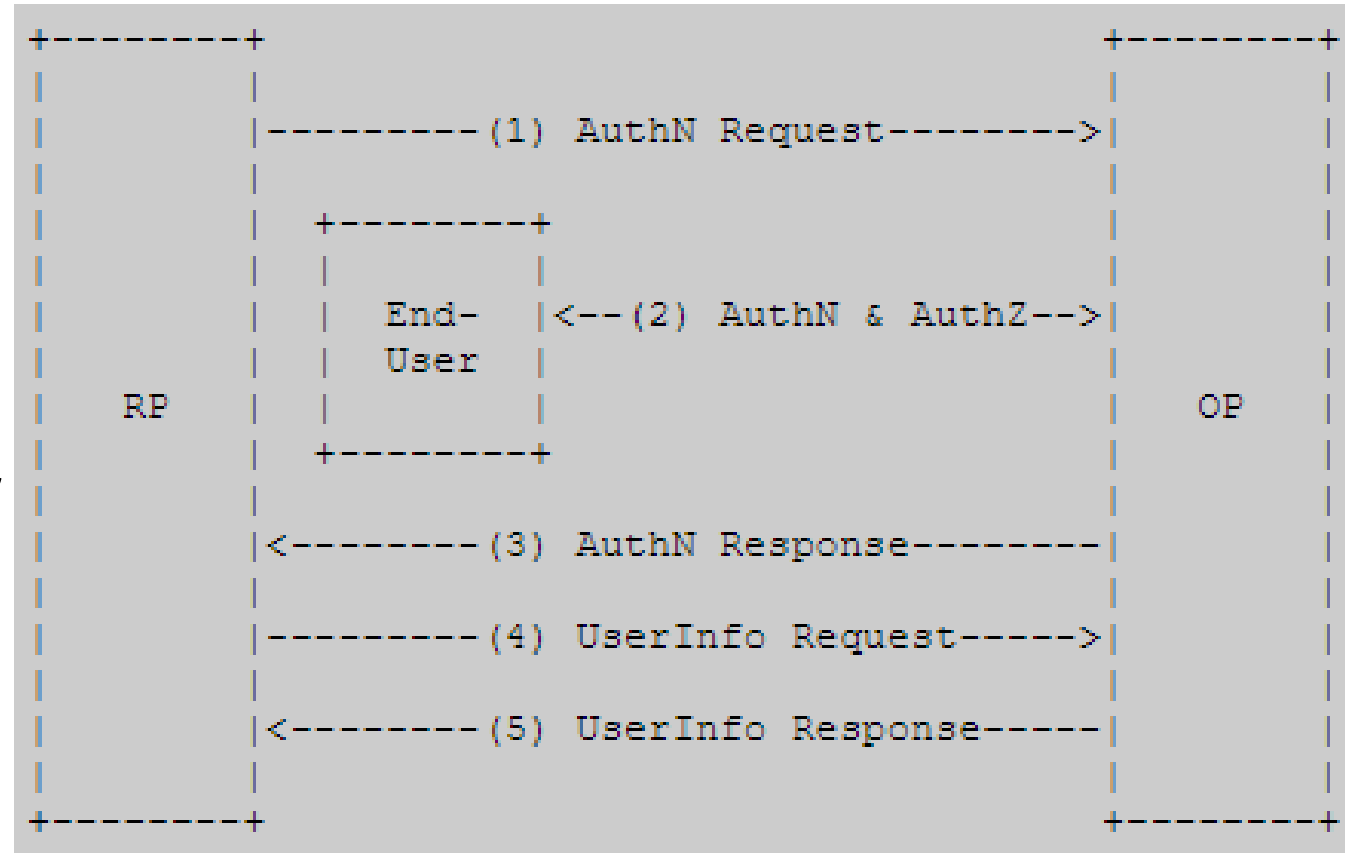
OpenID Connect Endpoints

- Authorization Endpoint (/authorize)
- Token Endpoint (/token)
- UserInfo Endpoint (/userinfo)
- https://openid.net/specs/openid-connect-core-1_0.html
- https://openid.net/specs/openid-connect-discovery-1_0.html



OpenID Connect Authentication Flows

- Authorization Code Flow -> "code"
- Implicit Flow -> "id_token" or "id_token token"
- Hybrid Flow -> "code id_token" or "code token" or "code id_token token"

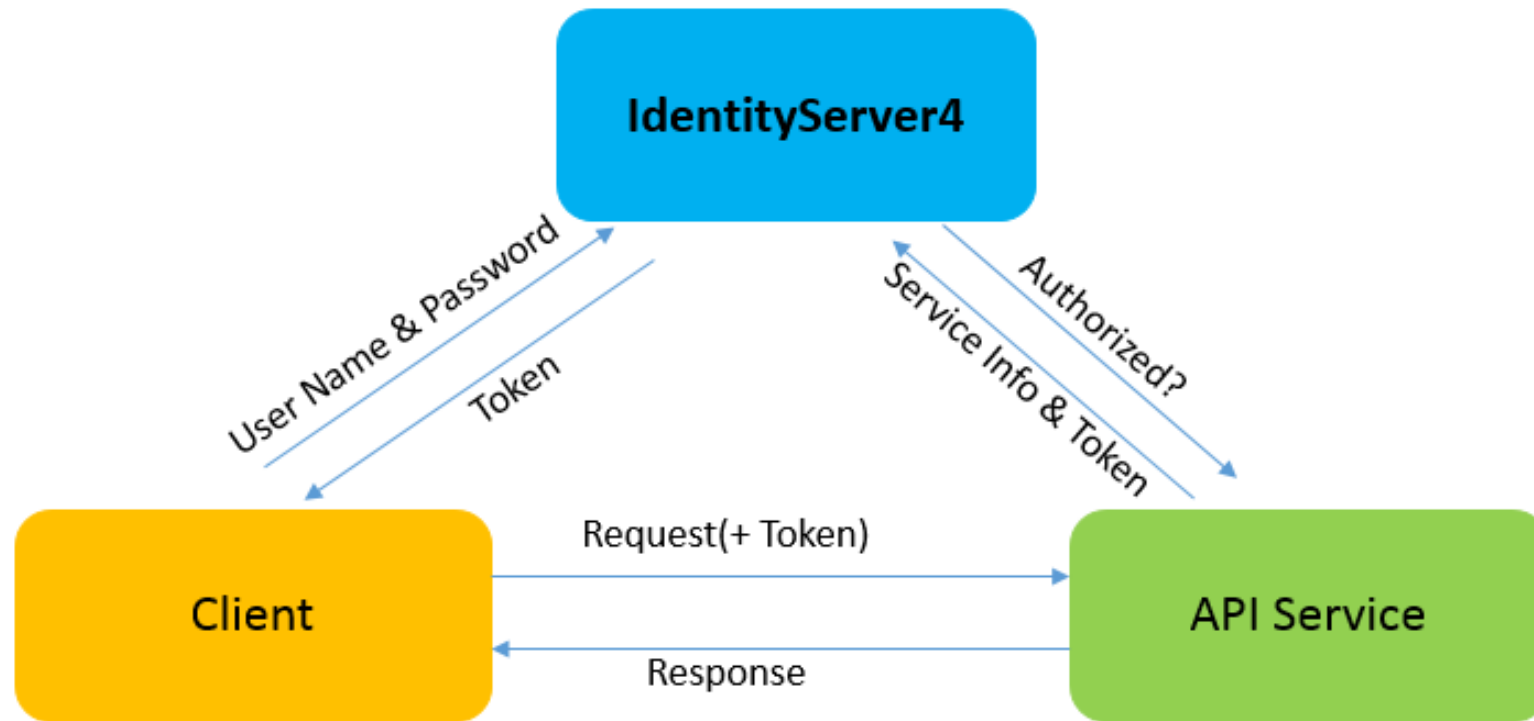


What is Identity Server 4 ?

- Open source framework
- OpenId Connect and OAuth2 implementation
- Centralized security system
- Issues security tokens to clients.
- Features – protect your resources



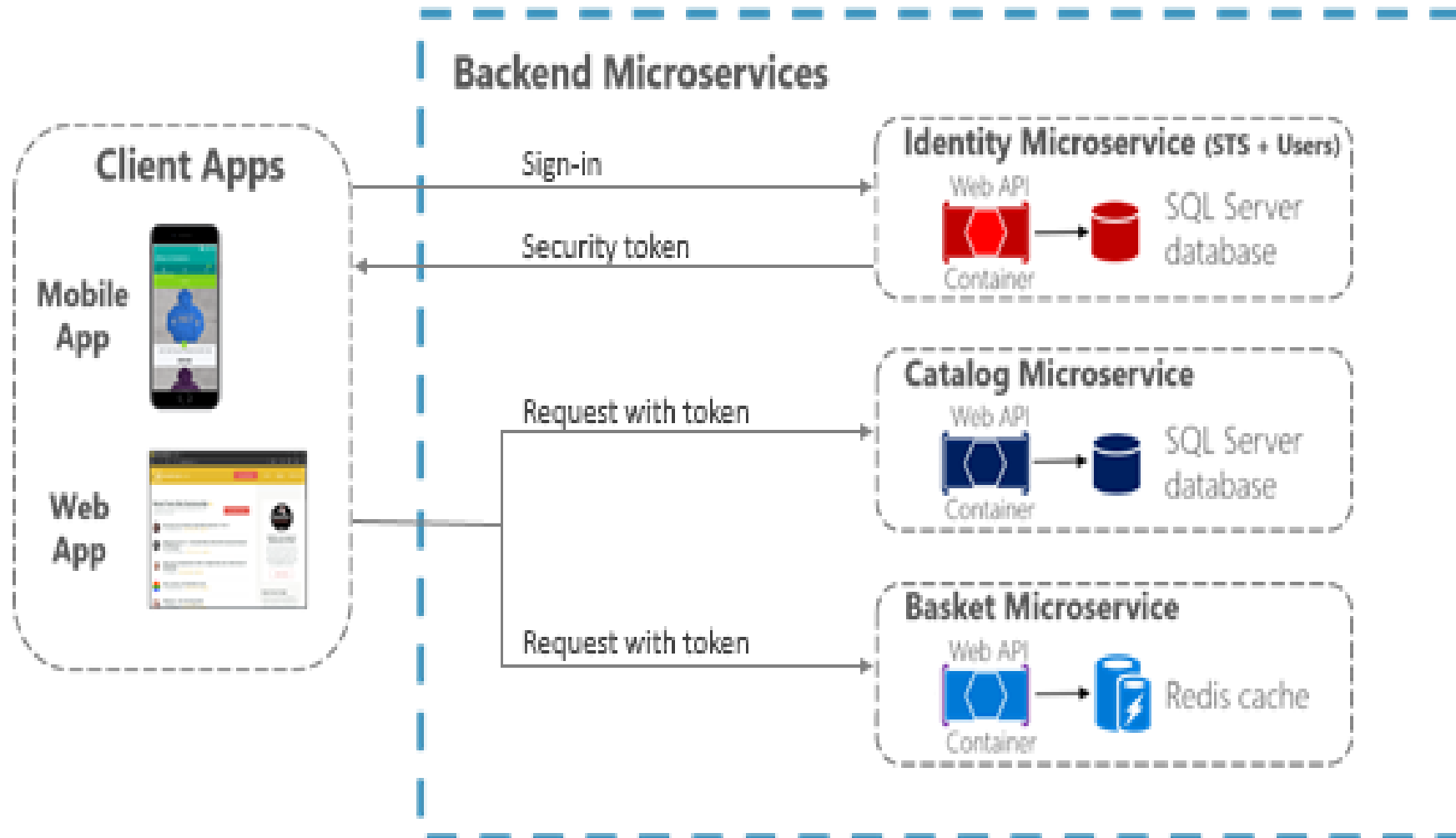
identity
SERVER



Identity Server 4 Terminologies

- Client Applications
- Resources
- Identity Token
- Access Token

Identity Server 4 in Microservices World



END OF SECTION

Next Section

BUILDING API RESOURCE

MOVIE.API ASP.NET WEB API PROJECT

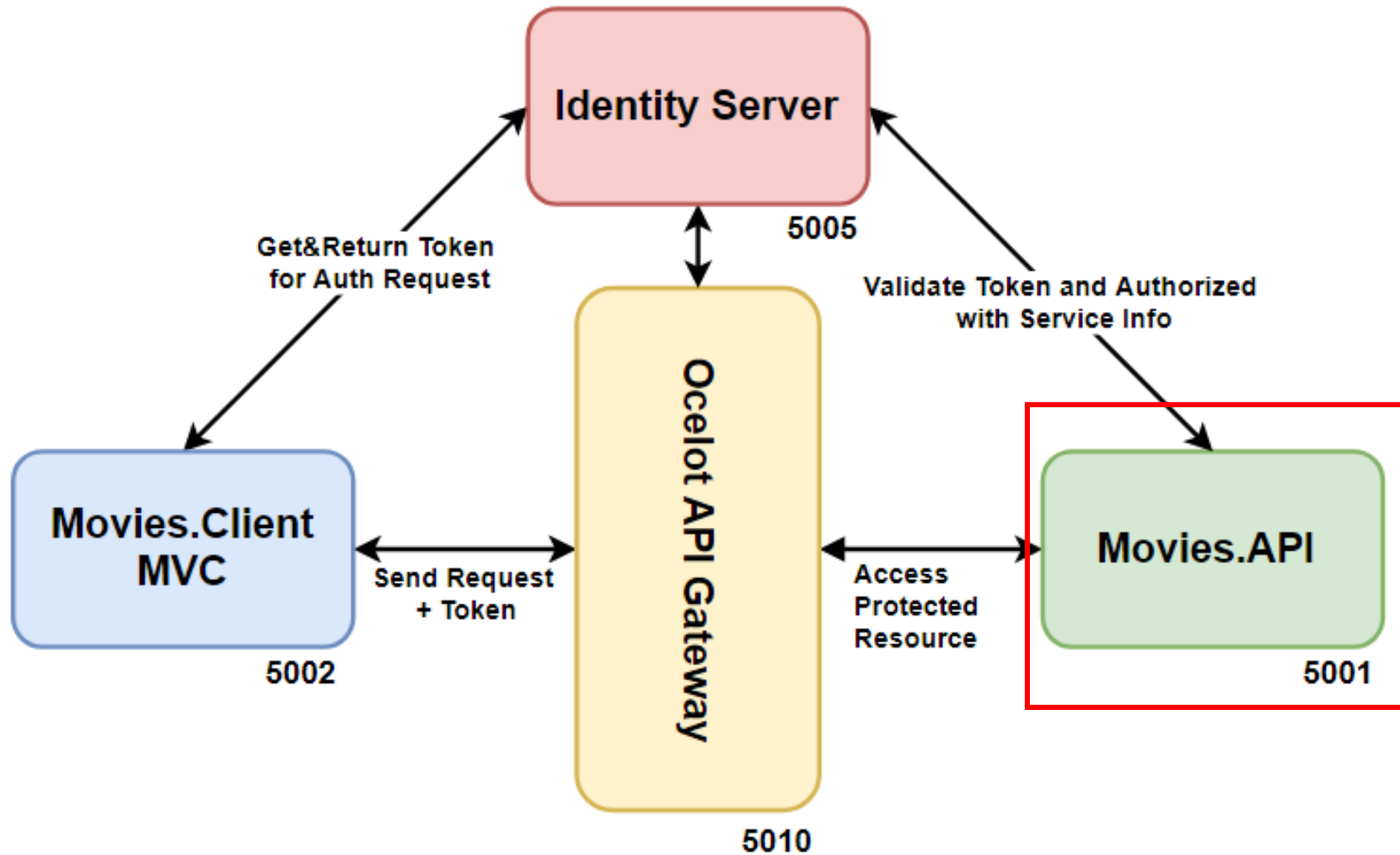


Section 2

Building API Resource

for building MOVIE.API Asp.Net Web Api Project

Big Picture



Movies.API Project

Method	URI	Operation	Description	Request body	Response body
GET	/api/movies	Read	Get all movie records	None	Array of movie records
GET	/api/movies/{id}	Read	Get a movie record by ID	None	Movie record
POST	/api/movies	Create	Add a new movie record	Movie record	Movie record
PUT	/api/movies/{id}	Update	Update an existing movie record	Movie record	None
DELETE	/api/movies/{id}	Delete	Delete a movie record	None	Movie record

- CRUD operations with Entity Framework InMemory Implementation

DEMO

Section 2 - Building API Resources



END OF SECTION

Next Section

BUILDING IDENTITY SERVER

AUTHENTICATION MICROSERVICES

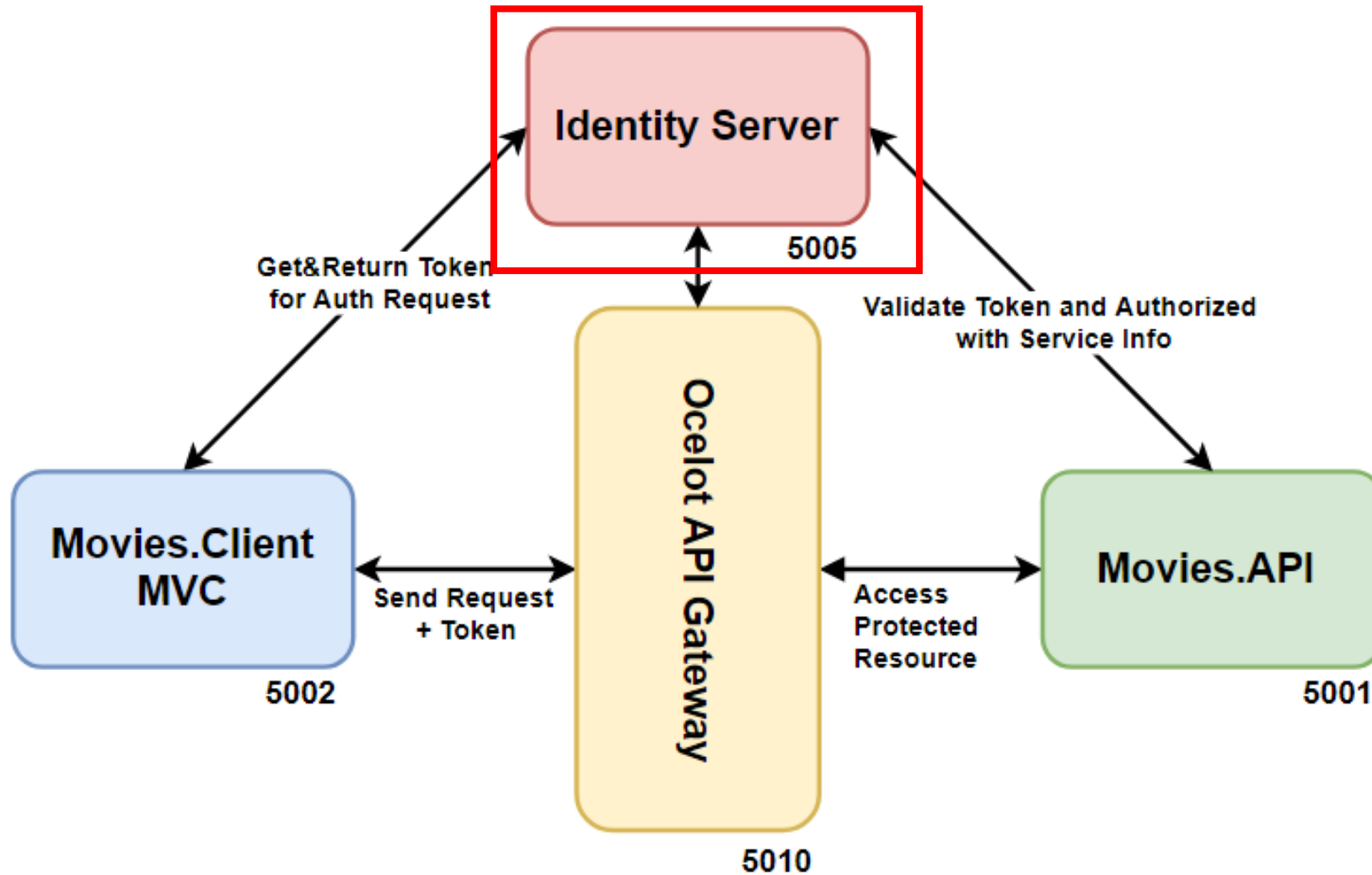


Section 3

Building Identity Server 4

for building Standalone Authentication Microservices

Big Picture



DEMO

Section 3 - Building Identity Server 4



IdentityServer4 Clients, Resources, Scopes and TestUsers

- ApiResources
- ApiScopes
- Clients
- IdentityResources
- TestUsers
- DeveloperSigningCredential



identity
SERVER

END OF SECTION

Next Section

PROTECTING MOVIE API WITH

USING IDENTITYSERVER 4 JWT TOKEN

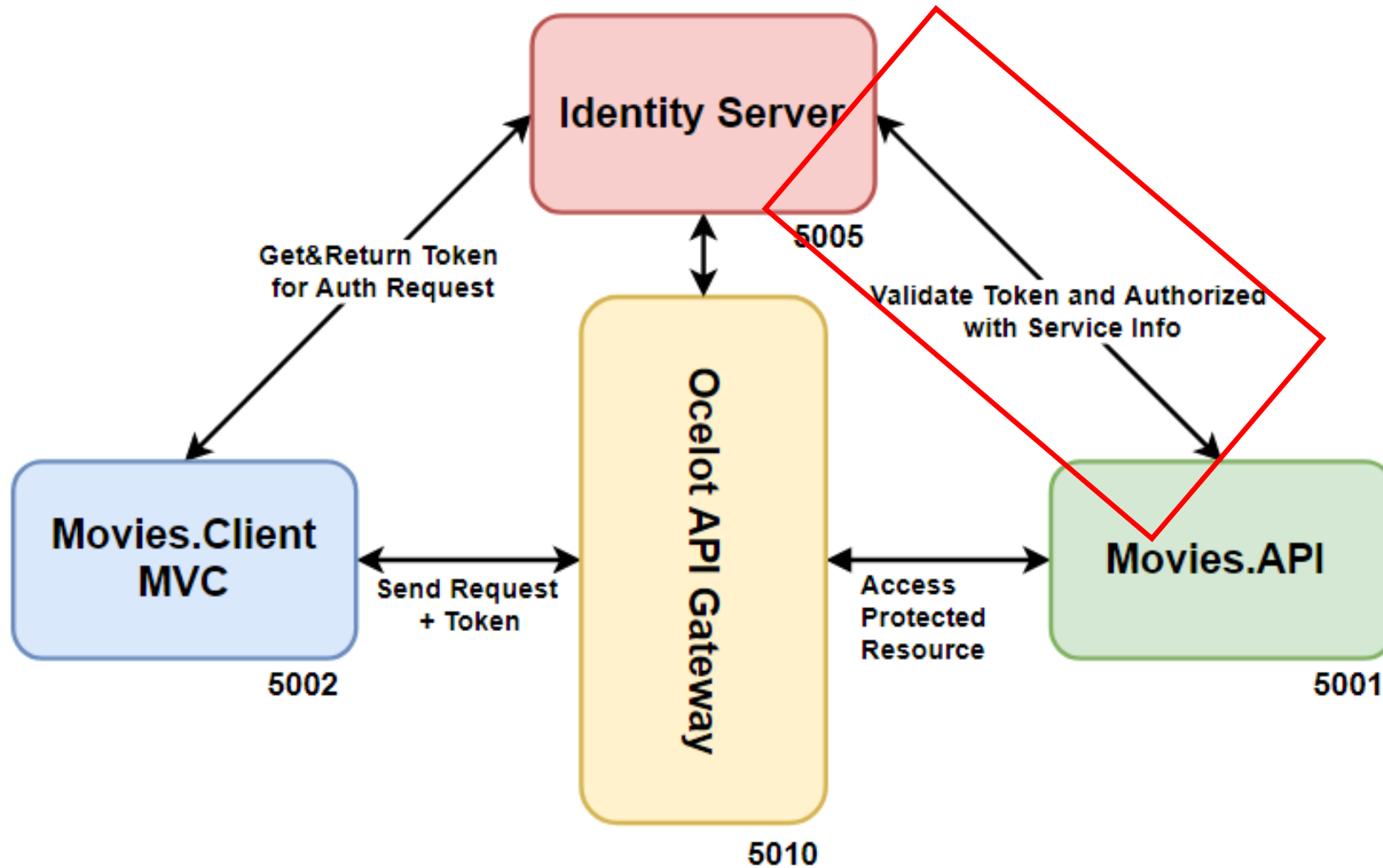


Section 4

Protecting Movie.API

With Using Identity Server 4 JWT Token

Big Picture



DEMO

Section 4 – Protecting Movies.API

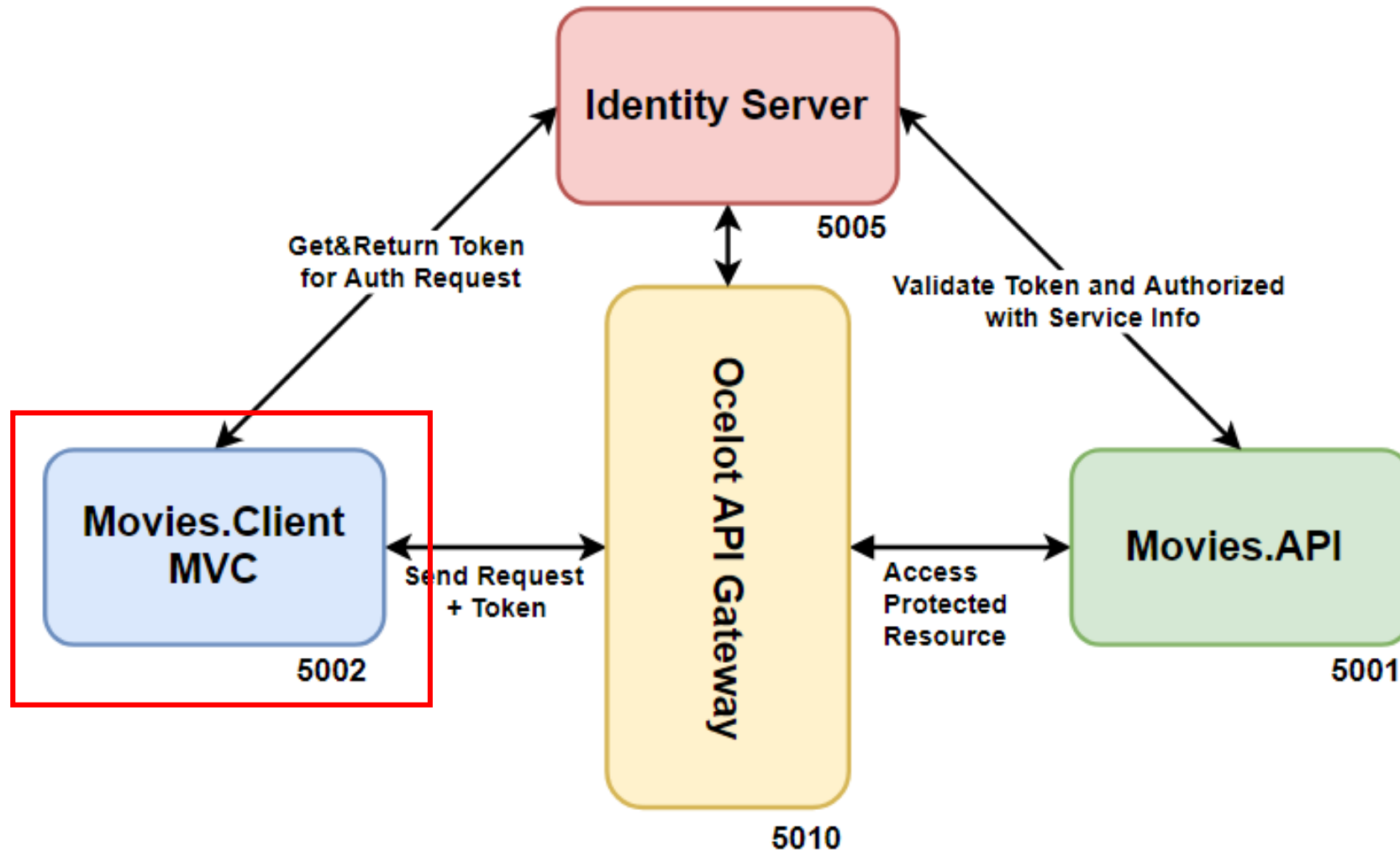


Section 5

Building MVC Client App

For Consuming Api Resource – Movies.Client Asp.Net MVC Web Project

Big Picture



DEMO

Section 5 – Building MVC Client



END OF SECTION

Next Section

OpenID Connect with Identity Server

For Interactive MVC Movie Client Microservice

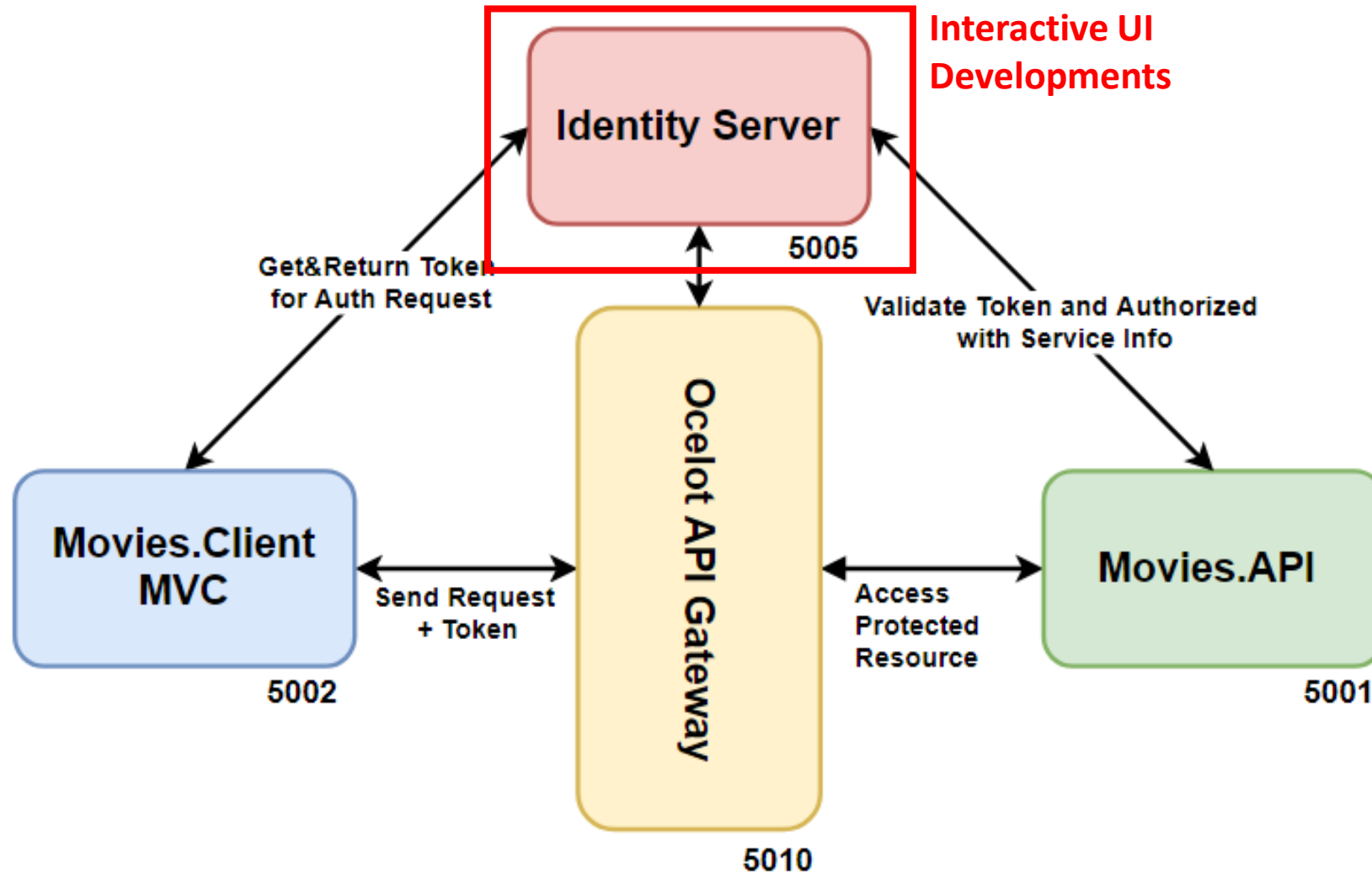


Section 6

OpenID Connect with IS4

For Interactive MVC Movie Client Microservice

Big Picture



DEMO

Section 6 – OpenID Connect with IS4



END OF SECTION

Next Section

Identity Server4 OpenId Connect Integration

For Interactive MVC Movie Client Microservice

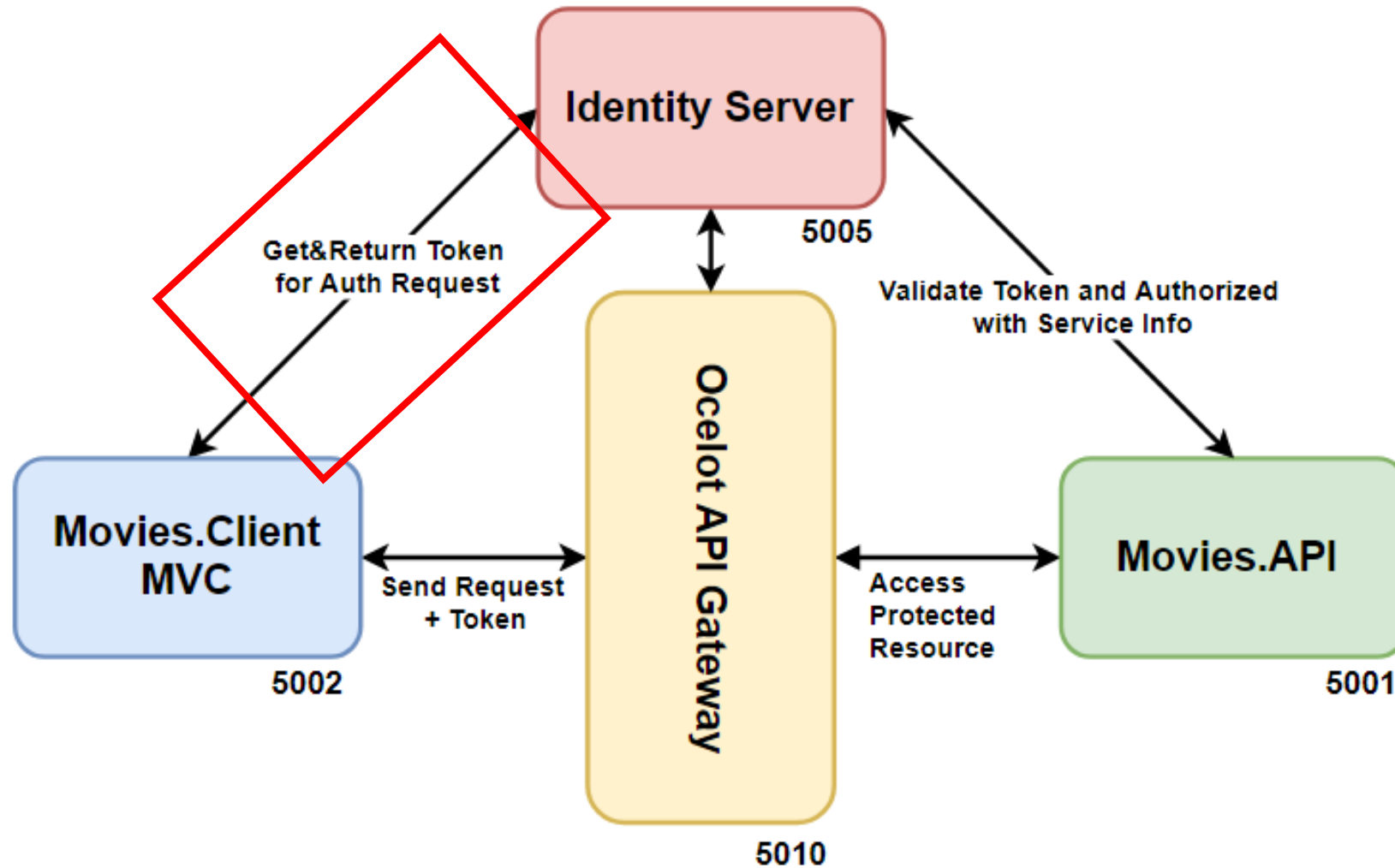


Section 7

Identity Server OpenID Connect Integration

For Interactive MVC Movie Client Microservice

Big Picture



DEMO

Section 7 – IS4 OpenID Connect Integration



END OF SECTION

Next Section

Consume Protected API From Movie.API Project

With HttpClientFactory into Movie.Client



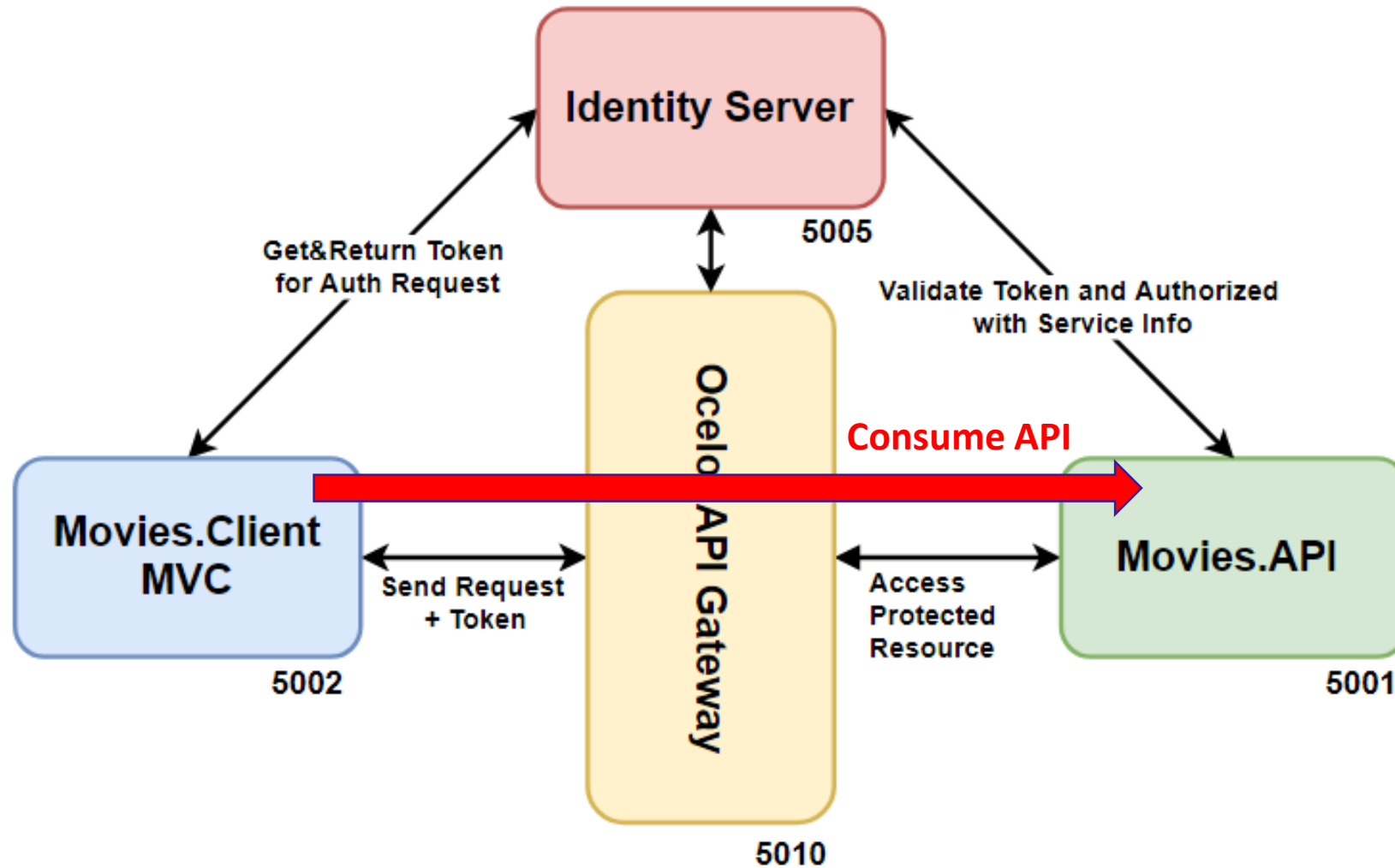
Section 8

Consume Protected

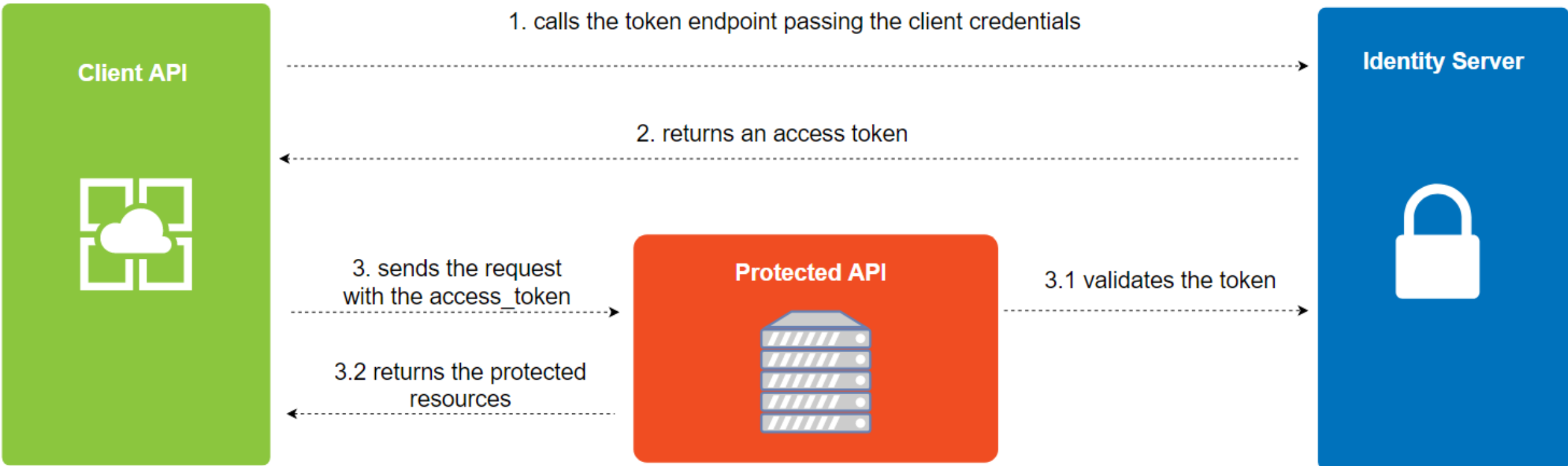
Movie.API

With HttpClientFactory From Movie.Client

Big Picture



Consume Protected API



CRUD Operations on Movies.Client

Method	URI	Operation	Description	Request body	Response body
GET	/api/movies	Read	Get all movie records	None	Array of movie records
GET	/api/movies/{id}	Read	Get a movie record by ID	None	Movie record
POST	/api/movies	Create	Add a new movie record	Movie record	Movie record
PUT	/api/movies/{id}	Update	Update an existing movie record	Movie record	None
DELETE	/api/movies/{id}	Delete	Delete a movie record	None	Movie record

- CRUD operations will be http calls with required token from Protected Movies.API microservices.

DEMO

Section 8 – Consume Protected Movie.API



END OF SECTION

Next Section

Hybrid Flow of Identity Server

Secure Interactive MVC Client (OpenID) and API
Resources (OAuth) with Together



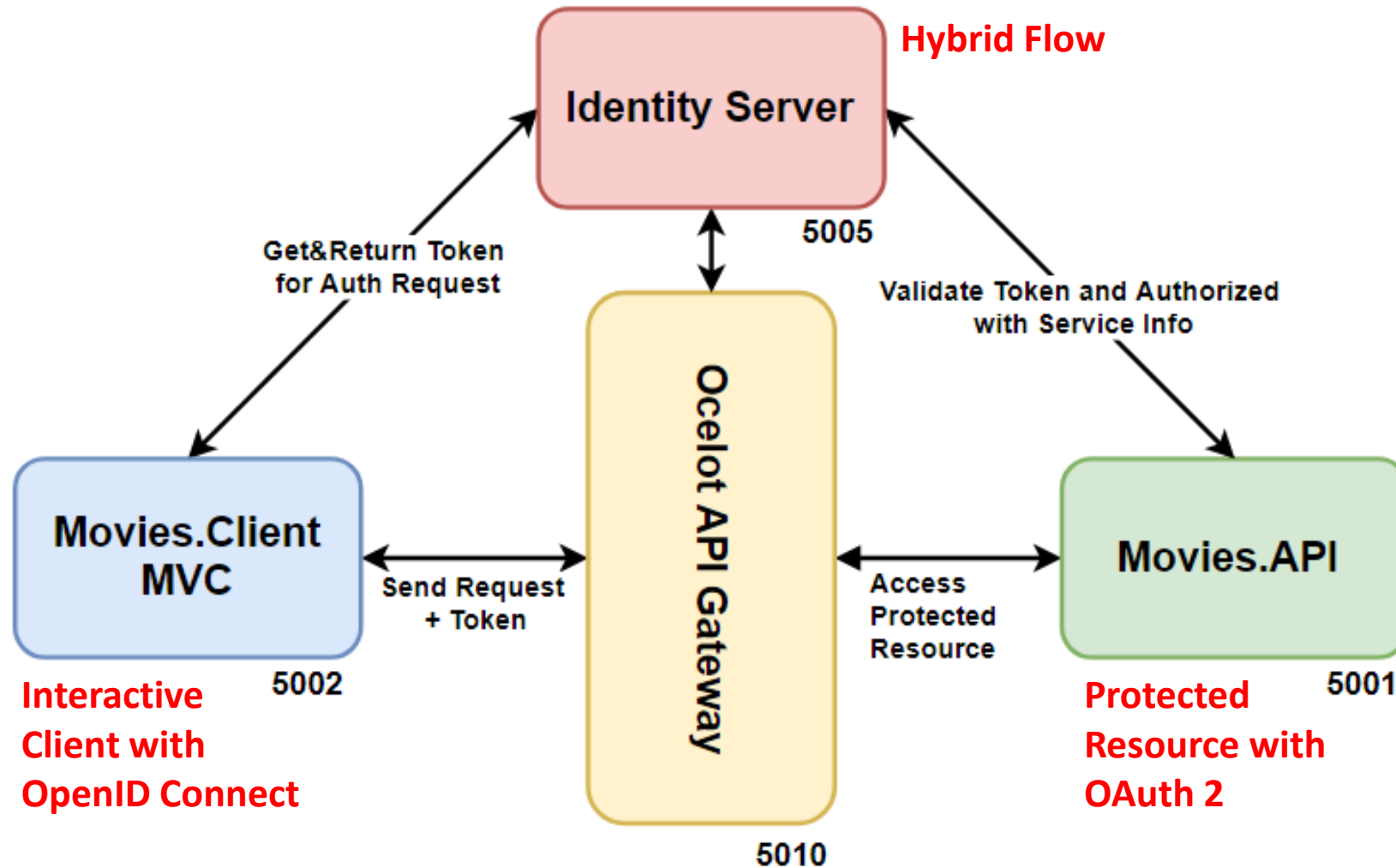
Section 9

Hybrid Flow of Identity

Server

Secure Interactive MVC Client (OpenID) and API Resources (OAuth) with Together

Big Picture



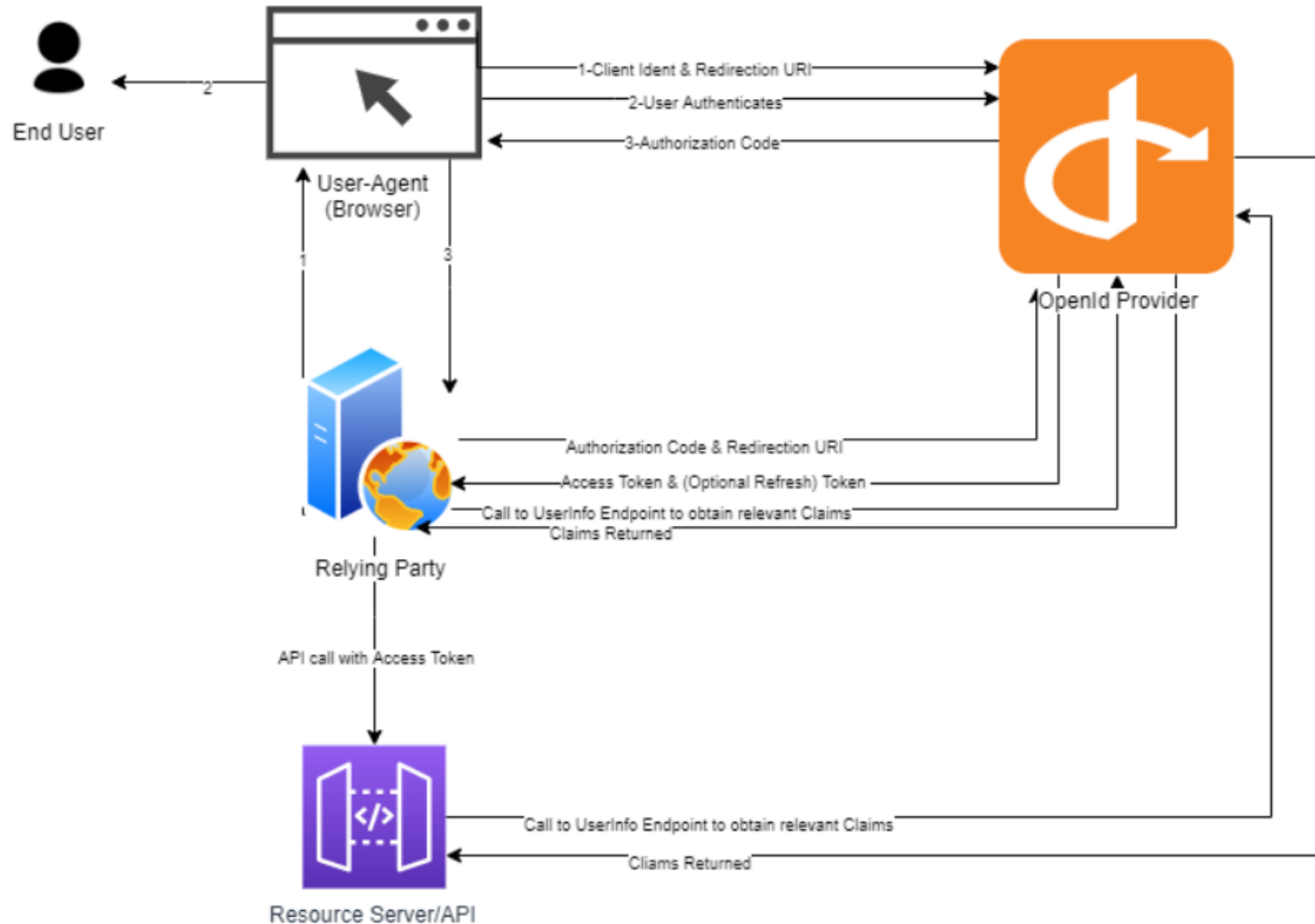
OIDC Authentication Flows

- **Authorization Code Flow** -
> "code"
- **Implicit Flow** ->
"id_token" or "id_token token"
- **Hybrit Flow** -> "code
id_token" or "code token" or
"code id_token token"

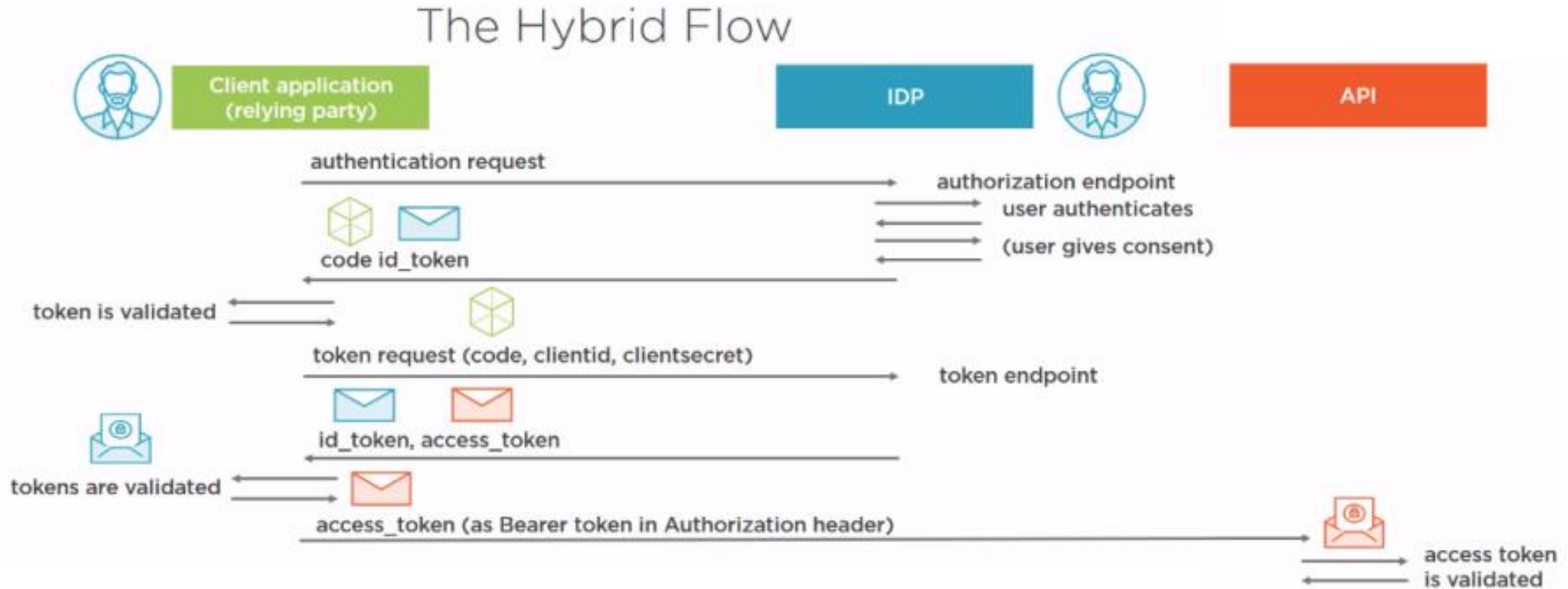


identity
SERVER

Authorization Code Flow (OIDC)



Hybrid Flow (OIDC)



DEMO

Section 9 – Hybrid Flow on Identity Server



END OF SECTION

Next Section

Claim Based Authorization

Secure Interactive MVC Client (OpenID) and API
Resources (OAuth)

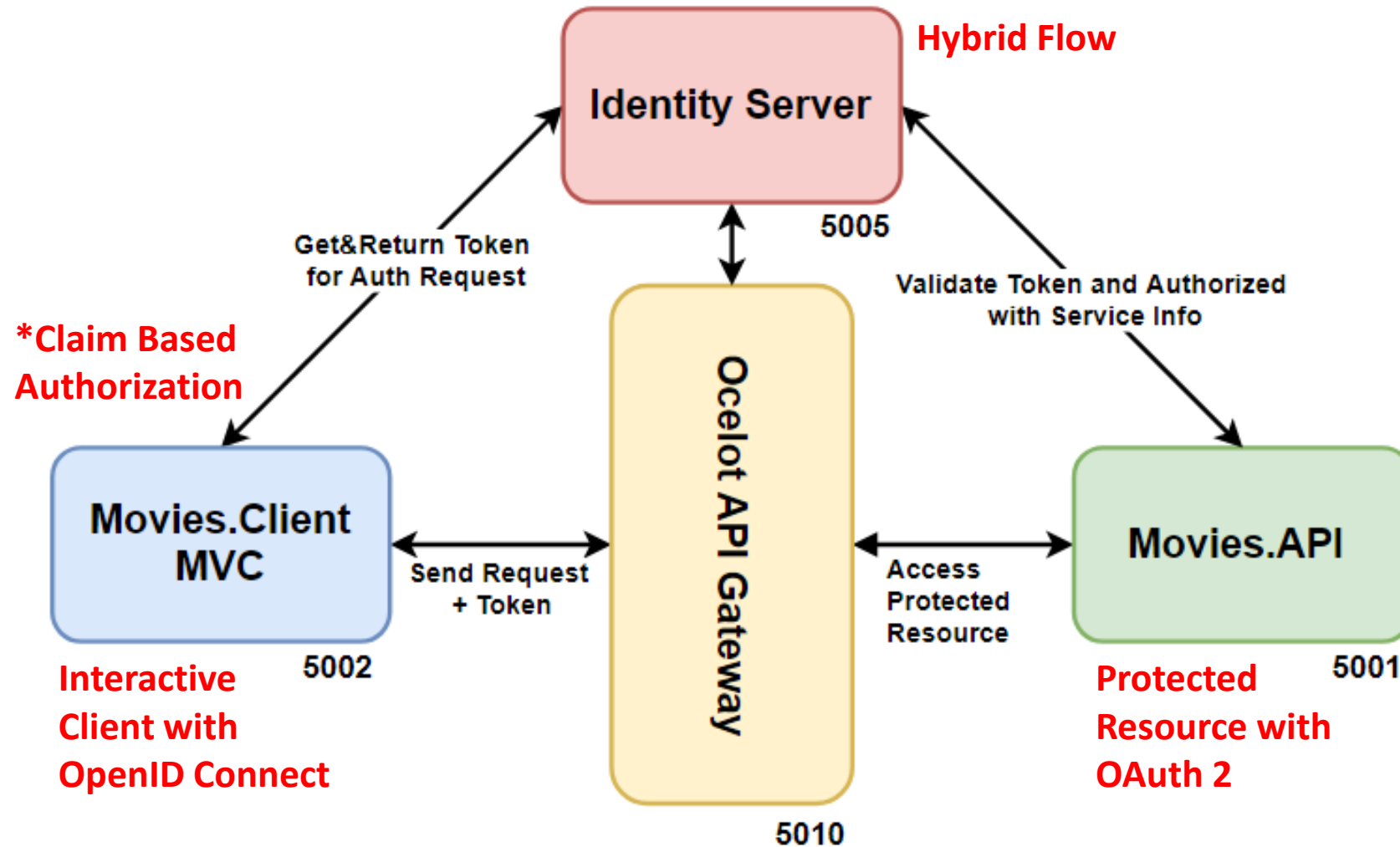


Section 10

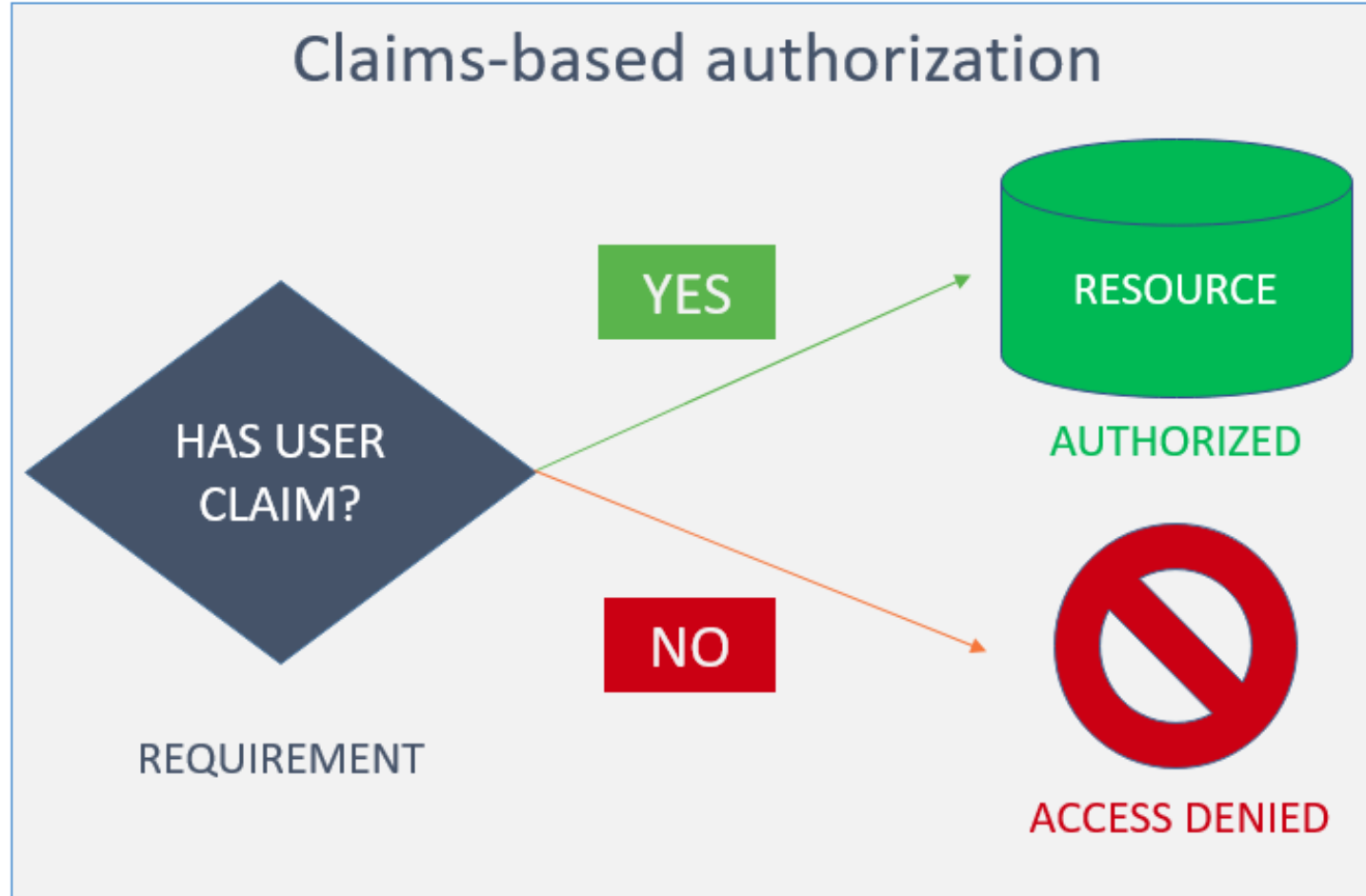
Claim Based Authorization in IS4

Secure Interactive MVC Client (OpenID) and API Resources (OAuth)

Big Picture



Claim Based Authorization



DEMO

Section 10 – Claim Based Authorization



END OF SECTION

Next Section

Ocelot API Gateway Implementation

For Interactive MVC Client (OpenID) to Interact
with Identity Server and Carry Token

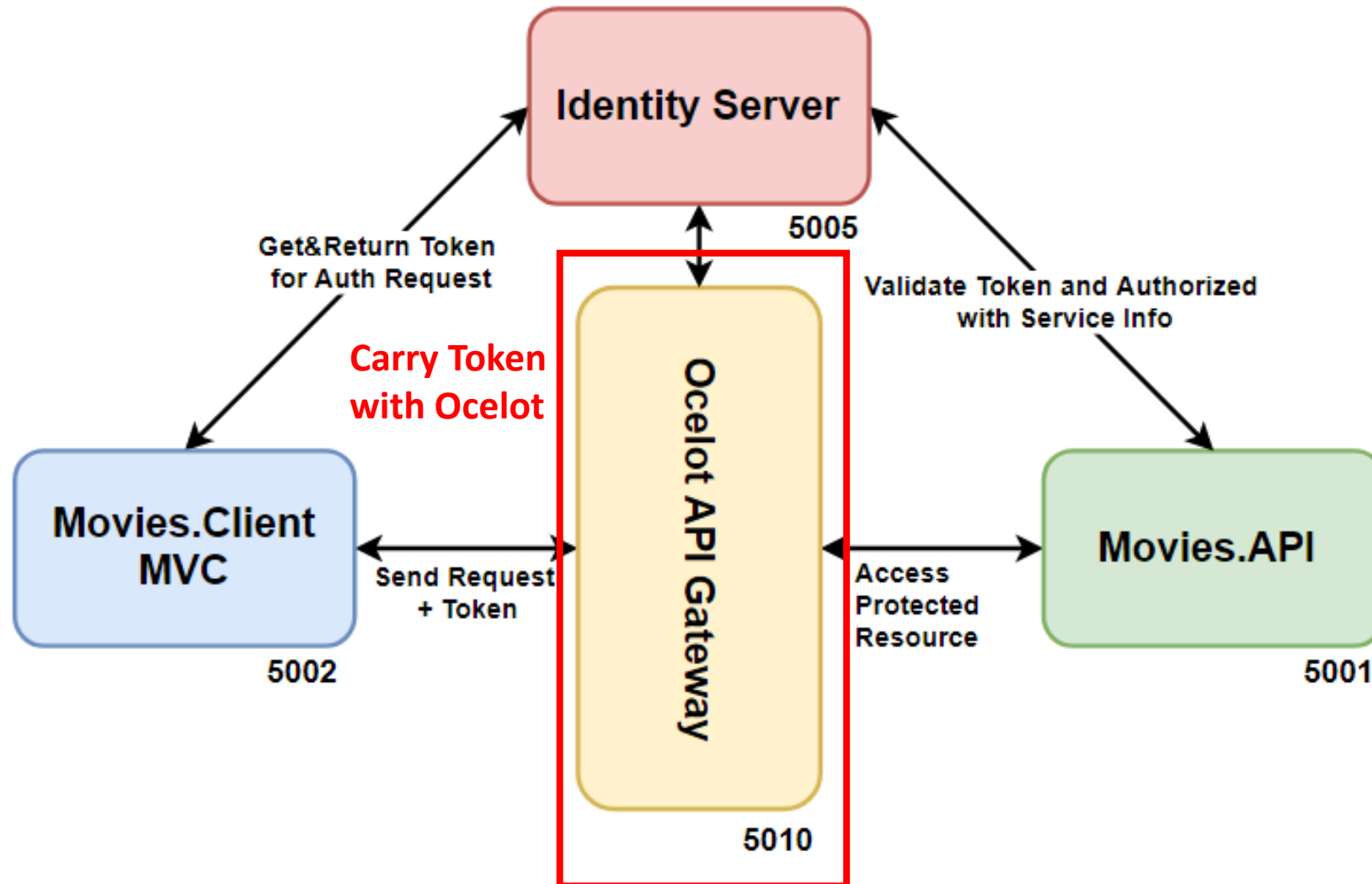


Section 11

Ocelot API Gateway Implementation

For Movies MVC Client to Interact with Identity Server and Carry the Token

Big Picture



DEMO

Section 11 – Ocelot API Gateway Impl



END OF SECTION

Next Section

Ocelot API Gateway Implementation

For Interactive MVC Client (OpenID) to Interact
with Identity Server and Carry Token



Section 12

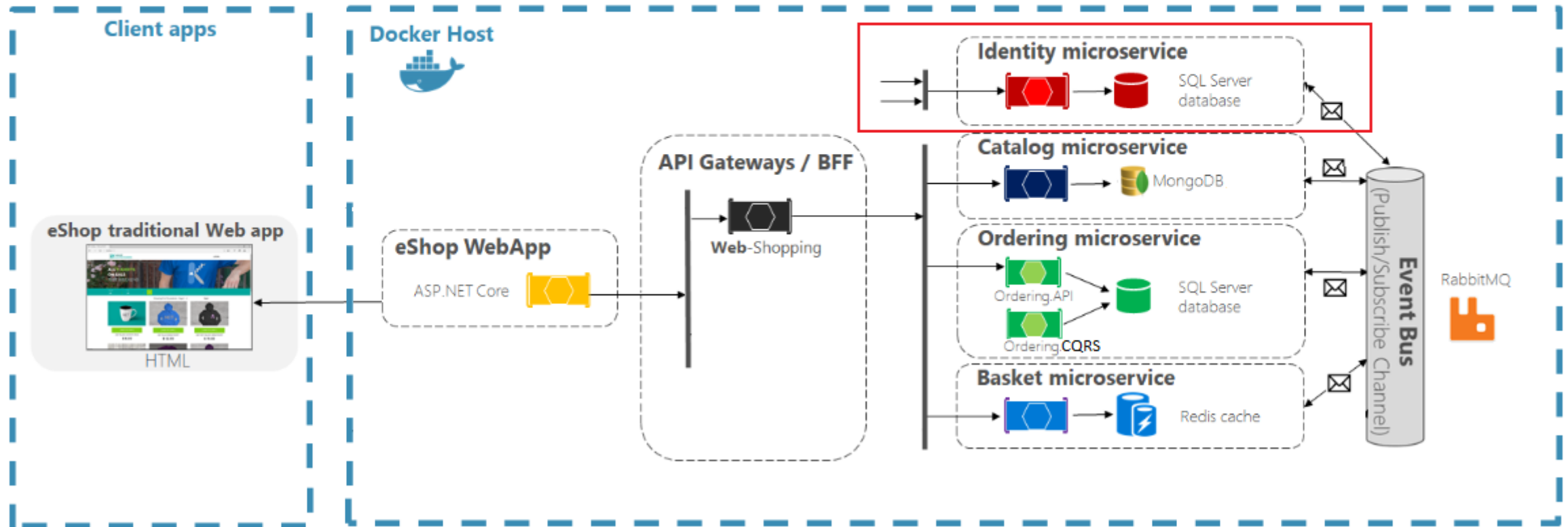
Secure Existing Microservices Application

With adding Identity Server Integration over the Ocelot and APIs


Identity Server in Microservices Architecture





aspnetrun-microservices Environment Architecture





Source Code of Microservices Architecture


 [aspnetrun / run-aspnetcore-microservices](#)


 Sponsor


 Used by ▼ 5


 Unwatch ▼ 6


 Unstar 92


 Fork 11


 Code


 Issues 0


 Pull requests 0


 Actions


 Projects 0

 Wiki

 Security 0

 Insights

 Settings

Building Microservices on .Net platforms which used Asp.Net Web API, Docker, RabbitMQ, Ocelot API Gateway, MongoDB, Redis, SqlServer, Entity Framework Core, CQRS and Clean Architecture implementation. Download Microservices Architecture and Step by Step Implementation on .NET Book -> <https://aspnetrun.azurewebsites.net/M...> 

aspnet-core

docker

ocelot-gateway

micorservices

aspnetcore-microservices

rabbitmq

mongodb

redis

sql-server

cqrs-pattern

clean-architecture

event-sourcing

eventbus

event-driven

microservices-architecture

aspnet-web-api


swagger


rest-api


api-gateway


mediator-pattern


[Manage topics](#)


 87 commits

 1 branch

 0 packages

 0 releases

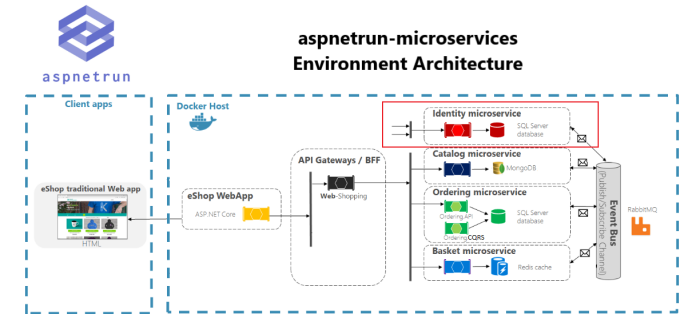
 1 contributor

 MIT

<https://github.com/aspnetrun/run-aspnetcore-microservices>

TODO List of Identity Microservices

- Create Identity Server Microservice into Reference Microservice Application
- Create Clients, Identity Resources and Testusers
- Create User Interface for Identity Server
- Set Authentication for Shopping MVC Interaction Client
- Add Login / Logout Button into Shopping Web Client Application



Thanks!

Follow me on github

<https://github.com/mehmetozkaya>

Follow me on twitter

<https://twitter.com/ezozkme>

Follow me on medium

<https://medium.com/@mehmetozkaya>

Send mail me anything you can ask

ezozkme@gmail.com

Check my other courses on udemy:

<https://www.udemy.com/user/ff0e5c8c-dd71-443e-be0a-e73ba821f7d7/>

